Integrating Decision Support, based on the Arden Syntax, in a Clinical Laboratory Environment

Bo Johansson * and Yngve Bergqvist **

* Department of Medical Informatics, Linköping University,
S-581 83 Linköping, Sweden
** Department of Clinical Chemistry, Falun Central Hospital,
S-791 82 Falun, Sweden

ABSTRACT

A clinical decision support system prototype have been developed in the clinical laboratory environment. The knowledge base consists of Medical Logic Modules, written in the Arden Syntax, and the work describes how these modules can be written, evoked and executed in a system, that is integrated with a laboratory information system, and facilitate real time validation of laboratory data. Tools and methods for building a decision support system are described and design aspects, such as database access, system validation and platform independence, are discussed.

INTRODUCTION

Decision Support Systems (DSS) are applied to the medical domain to emulate the behaviour of human experts. In clinical laboratories, DSS are used to improve internal efficiency and to assist the users of the laboratory. Quality control systems and systems for interpretation of laboratory data are available today [1] [2] [3].

Medical knowledge in a DSS can be represented in two different ways, rule-based and table-based. The knowledge base in a rule-based system, such as Mycin [4] and HELP [5], consists of rules written in some procedural language. Not only the medical knowledge but also the critiquing knowledge [6], describing how to apply the medical knowledge in a specific situation, is often stored in the rules. In table-based systems, such as Internist-1/QMR [7], knowledge is declarative and can be represented by frame hierarchies. Table-based systems make is easier to separate between medical and critiquing knowledge.

In this paper we demonstrate how a standardised language for building rule-based knowledge bases, Arden Syntax [8], is chosen for a DSS in the clinical laboratory environment. Tools and methods for handling knowledge modules have been developed [9] [10] [11] and are used in the construction of a DSS prototype. A domain expert can add knowledge to a knowledge base, define how and when to evoke the knowledge module and select the appropriate action that will take place for certain conditions specified in the knowledge module.

BACKGROUND

Arden Syntax is a language standard for representation of health knowledge in medical information systems, where the knowledge is stored as independent Medical Logic Modules, MLMs [12]. Each MLM contains sufficient knowledge to make a single decision. Arden Syntax allows knowledge bases to be shared among institutions working in the same medical domain. Several institutions have been involved in the development of the standard, including LDS Hospital, Salt Lake City, USA, Linköping University, Linköping, Sweden, Erasmus University, Rotterdam, The Netherlands, and Columbia-Presbyterian Medical Center (CPMC), New York, USA. The syntax is derived largely from HELP [5] and CARE [13] and one goal is to make it easy to read and write by health personnel with little computer training. Some areas where knowledge can be represented with the Arden Syntax are contraindication alerts, management suggestions and data interpretation.

Both medical and critiquing knowledge are written in the MLM, where each module contains maintenance information, links to external patient data and logic that can make a single health decision. In figure 1 an example of a MLM is displayed. The knowledge module checks the analyte sodium to see if the current analytical result differs much from the previous result for the same patient. The MLM is data driven and is evoked when a sodium result is stored in the laboratory database. Note that the read statement in the data slot has an institution specific part, placed between curly brackets. Each institution that shares the MLM can change this part and add a mapping that is valid for their own database management system (DBMS). In this example a SQL query is selected in the read statement.

The objectives for the work presented here are to verify methods and tools for building a DSS based on MLMs and to demonstrate an integration of a DSS with an existing information system at a hospital. After the decision to integrate the DSS with a laboratory information system (LIS) at a clinical laboratory, one aim was to investigate if the DSS could give support in the validation process of laboratory test results.

One advantage with electronically stored patient data in the clinical laboratory is the possibility of automatic surveillance of test results. Close integration between the

0195-4210/92/$5.00 © 1994 AMIA, Inc.

DSS and the information system where patient data are stored (LIS, HIS or other) is necessary for access to databases since manual input of data is not practical.

```
maintenance:
    title:        Delta-check for Sodium;;
    filename:     Sodium_delta;;
    version:      1.0;;
    institution:  Dept of Clinical Chemistry, Falun Central
                  Hospital, Sweden;;
    author:       Bo Johansson;;
    specialist:   ;;
    date:         1992-09-15;;
    validation:   testing;;

library:
    purpose:      Delta-check for validation of Sodium results;;
    explanation: Compare current result with previous result for
                  same patient;;
    keywords:     Quality control; Delta check; Sodium;;
    citations:    Lacher DA, Connelly DP. Rate and Delta Checks
                  Compared for Selected Chemistry Tests. Clinical
                  Chemistry, 1988;34; 1966-1970;;

knowledge:
    type:    data-driven;;
    data:    current_value := read last {SELECT RESULT FROM
                 DAYREG WHERE ANALYTE="SODIUM"};
             previous_value := read last {SELECT RESULT FROM
                 ACKREG WHERE ANALYTE="SODIUM"};
             delta_limit := read last {SELECT DELTA_CH FROM
                 LIMITS WHERE ANALYTE="SODIUM"};
             sodium_storage := event {"STORAGE OF  SODIUM");
             action_dest := destination {"Printer","PRINTER_Q1");;
    evoke:   sodium_storage;;
    logic:   delta_percent_change := (( current_value -  previous_value )
             / previous_value )*100;
             if abs delta_percent_change > delta_limit then
               action_message:="Delta check warning (comparison with
                   previous value).  Change since last Sodium test is
                   ||delta_percent_change||"%, which is greater
                   than limit "||delta_limit||"% ";
               conclude TRUE;
             else
               conclude FALSE;
             endif;;
    action:  write action_message at action_dest;;
```

Figure 1. Example of a knowledge module written in the Arden Syntax. This Medical Logic Module (MLM) is used for validation of sodium test results.

DESIGN CONSIDERATIONS

Methods where MLMs, written with an Arden Syntax knowledge editor, are syntax checked and translated into the object oriented language C++ have been developed [9] [10] [11] and are followed in this implementation of a DSS in the clinical laboratory at Falun Central Hospital in Sweden. Tools have mainly been constructed in the UNIX environment [11] and a standard C++ compiler is used for compilation of the C++ modules. To achieve a flexible integration with the LIS, running on a PC platform under DOS, we decided to implement the DSS in the same environment.

One reason to build a DSS integrated to a LIS is surveillance of laboratory test results in the daily routine and to detect errors before results leave the laboratory. This is achieved by writing MLMs that perform calculations on test results, such as limit and trend checks, and send alerts for abnormal laboratory data.

To obtain a successful implementation of decision support to an existing LIS, the DSS need to be integrated with the LIS in a way that facilitates access to patient data in real time without manual input. Databases in the LIS must be shared with the DSS. Also action messages from the DSS, such as alerts, must be sent to the LIS. Triggers that evoke MLMs in the DSS are also sent from the LIS to the DSS.

The software need to be modularised with a database interface, that makes access to different databases easy to handle, and a flexible communication interface for different implementations of triggers and alerts. For validation of MLMs, a simulation and debugging environment is important to consider in the DSS design.

SYSTEM DESCRIPTION

A diagram of the LIS and the DSS is shown in figure 2. The diagram also shows the interaction between the two systems and how a domain expert adds knowledge to the DSS.

Laboratory Information System

At Falun Central Hospital, with 700 beds, the clinical laboratory, processing 800.000 tests per year, have used the Daltek Laboratory Data System [14] in production since 1989. The Daltek system is a LIS, running on IBM PC or compatibles under DOS operating system. IBM Token Ring local area network (LAN) with Novell Netware 3.11 as network operating system is used and laboratory patient data are stored in dBase compatible (Xbase) databases.

Decision Support System

For an optimal integration with the LIS, the computer running the DSS was placed in the laboratory network. An IBM PC compatible computer was selected and for DSS software development the Borland C++ compiler [15] was chosen together with the class library Codebase ++ [16] for table handling. For shared access to the LIS databases we selected the dBCIII Plus library [17], which is the same database library that is used by the LIS.

The Knowledge manager (KM) module in the DSS is responsible for acquiring knowledge from the domain expert and for building and maintaining knowledge bases. Also functions for creating DSS are included in the KM. One important part of the KM is the MLM editor [9], that will assist the domain expert in the development process of MLMs. In this work on the DOS platform a standard text editor, Microsoft Editor [18], is used for editing purposes. Another tool in the KM is the MLM compiler for syntax checking and C++ code generation. In the DOS version of the compiler only C++ code is generated, the syntax is checked on the UNIX platform. Included in the KM are also tools for verification and validation of MLMs. Verification is done in a simulator that can execute MLMs with simulated patient data, keyed in manually from the keyboard or retrieved from a test database. Validation is done to assure that the MLM works properly in an environment with real patient data. This is very important, especially when MLMs are imported from other institutions. In the verification and validation processes we use a standard debugger, included with the Borland C++ compiler.

Integration of LIS and DSS

The close integration between the DSS and the LIS makes it possible to access patient data needed by the knowledge modules at runtime. This is achieved by selecting a library for database management that supports shared access to the LIS databases.

Action messages from the DSS, such as alerts and interpretations, are logged and sent to the LIS. Destination for the action message can be specified in the MLM as a mail, a printer output or an update in the patient database.

Interaction between the two systems can be described in several steps, see below. Step numbers refer to the numbers in figure 2.

Step 1. An event occurs in the LIS , which can be the storage of a laboratory test result in the patient database.

Step 2. A trigger (event message) is sent from LIS to Event / Evoke manager in DSS. Our LIS database is not active and has no trigger capabilities, a separate process is responsible for sending the event message. The process supervises the LIS result database and detects all new laboratory test results. An event message is created for each new result stored in the LIS.

Step 3. Event / Evoke manager receives the event message and stores it in the event table. Information in the evoke table shows the MLM(s) that will be evoked for the current event and when they will be evoked. All MLMs are stored in an MLM table, sorted by evoking time. One event leads to several occurrences in the MLM table if the evoke slot in the MLM defines a periodic trigger.

Step 4. Inference Engine searches the MLM table for MLMs to evoke.

Step 5. When an MLM is evoked, it is retrieved from the Knowledge Base and executed.

Step 6. During execution of an MLM, patient data are requested by the MLM data slot. The requests are handled by the Database manager and data are retrieved from the LIS patient database.

Step 7. If an MLM action is present, the specified action is performed. Different actions are possible. Typical actions are sending a message, returning a value to a calling MLM and evoking another MLM.

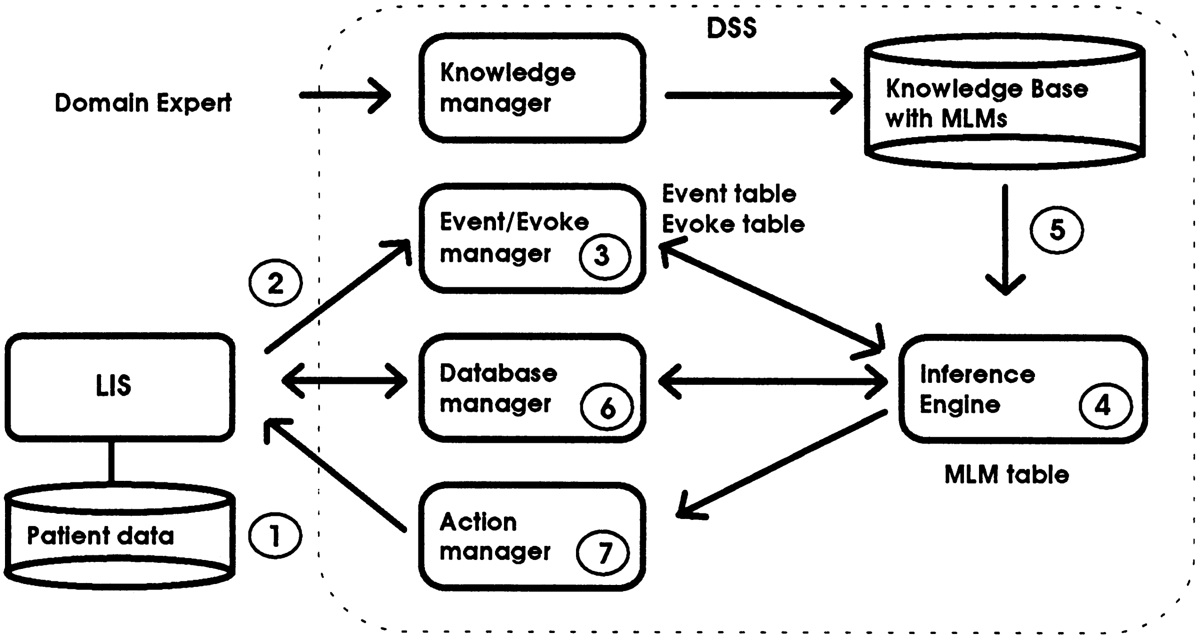

Figure 2. Laboratory decision support diagram.

All steps in the DSS process are logged which makes it possible to trace events that have occurred, evoked MLMs and actions that have taken place.

RESULTS

The software in the DSS is developed with an object oriented programming language and is built in separate modules, which have three different main functions; to handle evokes (when MLMs shall be executed), database access (data retrieval from the DBMS) and actions (how DSS give response). Database and communication interfaces have been constructed to make communication with databases and applications possible and to handle MLMs in the knowledge base.

Knowledge rules in MLM format for validation of patient results, by calculating delta check and rate check according to Lasher and Connelly [19], are included in the laboratory DSS. In the delta check method the current analytical result for a patient is compared with the previous result for the same patient. Rate check also relates the delta change to the time interval between the two tests. MLMs for limit checks of test results have also been written and run. A fully automatic surveillance of laboratory test results is achieved, no manual input by laboratory personnel is necessary. Alerts, created if the specified conditions are fulfilled, can be sent in real time to different destinations, such as screen, file, printer and electronic mailbox. The MLM in figure 1 shows an action message that is directed to a specific printer queue by defining a destination in the data slot and referring to that destination in the action slot.

Integration of the DSS has only a minor influence on the performance of the LIS since the DSS runs on a separate workstation in the LAN and only interacts with the LIS databases.

Validation of MLMs is very important, both technical and medical aspects. Before MLMs are moved into production they must pass a validation process. Validation is done especially of the knowledge category in the MLM, mainly the data, logic and action slots. One way we practise to verify that an MLM works as intended is to use a simulator, where the keyboard or a test database is used as input for requested data in the data slot. This makes it possible to run the MLM with known data and verify that the logic and action parts are correct. Further validation can be necessary to investigate the logic section or if a data operator doesn't work properly. Then the MLM is run in a debugger that can step through the code and trace possible errors in the logic or in some C++ method. Since the generated C++ code and the MLM code have the same structure, it is easy to follow the MLM when debugging the C++ code. A test period follows where the MLM accesses real patient databases and actions are examined and verified, then the MLM can be put into production. The validation process is also important when sharing MLMs with other institutions.

DISCUSSION

Adopting the Arden Syntax as a standard when building knowledge bases at clinical laboratories, for quality control, data interpretation etc., makes it easier to share knowledge bases between laboratories. Methods and tools to manage knowledge bases written in the Arden Syntax are available today on different computer platforms and integration to an existing laboratory information system can be made without major effort with the described method. When the DSS is built in modules the system can be transferred to different environments keeping the same structure, only modifications in the interfaces are necessary for access to patient databases and for communication of e.g. triggers and alerts. On the new DSS platform, a C++ compiler together with drivers for databases and communication are required. Standard SQL interfaces for database access, such as Microsoft's Open Database Connectivity (ODBC), are interesting for transparent access to different databases [20]. Data mapping in the Arden Syntax is not yet a standard, but institution specific. In this implementation we write a SQL query as data mapping to make porting to other platforms and sharing with other institutions easier.

Tools are not yet available for all platforms. In the described DOS system we have no advanced MLM editor for knowledge modules, only a standard text editor. Syntax checking of MLMs is still done on UNIX platform, our compiler is not yet implemented under DOS. Further development of the PC based DSS will take place under Windows, where the event driven programming facilities and a graphical user interface will be effective. Knowledge bases could then be implemented as dynamic link libraries (DLL). Large knowledge bases in the DOS version can be managed with overlays if the 640 kB limit is exceeded. Multiple Inference engines, with their own knowledge base, is another solution for large knowledge bases. The Event/Evoke manager then routes the events to different evoke tables, one for each Inference engine, and the DSS can be split into several processes in one or more physical computers.

Besides validation of laboratory data, interpretation of test results is an area where decision support is applied. Interpretation may not be a task for the clinical laboratory, but should be done close to the patient at the clinical department where other important patient data needed for interpretation, such as diagnosis and drug history, are available in the computerised medical record.

CONCLUSION

The work shows that our methods and tools for building DSS, based on the Arden Syntax, can be used for development of a DSS in a clinical laboratory environment. Integration of the DSS with a LIS, which includes real time access to patient databases and communication of trigger and alerts, has been done with success. Furthermore, support tools for writing and validating knowledge modules are presented.

MLMs for validation of laboratory test results have been written and tested in production. This has shown that alerts are created automatically, with no delay, for abnormal test results, and sent to the LIS. More work has to be done to adjust limits, used in the knowledge rules for detection of abnormal results, that give a balance between false positive and false negative alerts for the different analytes.

Acknowledgements

This work was supported by a grant from the Johan Gottlieb Gahn Academy and by the Commission of European Communities as part of the HELIOS project (AIM A2015). Partners in the HELIOS project are Broussais University Hospital (France), Cap Gemini Sogeti (France), DKFZ Heidelberg (Germany), Geneva State Hospital (Switzerland), Uppsala University (Sweden), Linköping University (Sweden) and DEC.

Many thanks to the clinical laboratory at Falun Central Hospital for their positive attitude to this work and for access to their LIS.

Reference

[1] O'Moore RR. Decision Support based on Laboratory Data. Meth. Inform. Med. 1988;27 187-190.

[2] Valdiguie PM, Rogari E, Philippe H. VALAB: Expert System for Validation of Biochemical Data. Clinical Chemistry;1992;38;83-87.

[3] Boran G, Eldridge P, Nolan J et Al. A decision support tool for laboratory medicine based on automatic knowledge acquisition. In O'Moore R, Bengtsson S, Bryant JR et Al. Lecture Notes in Medical Informatics, Berlin, Germany, Springer-Verlag, 1990; 40; 329-333.

[4] Shortliffe E. Computer-Based Medical Consultations: MYCIN. New York: Elsevier / North Holland, 1976.

[5] Pryor TA, Gardner RM, Clayton PD, Warner HR. The HELP system, In Blum BI ed., Information Systems For Patient Care, New York: Springer-Verlag, 1984; 109-128.

[6] Van Der Lei J, Musen MA. A model for Critiquing Based on Automated Medical Records. Computers and Biomedical Research, 1991, 24, 344-378.

[7] Miller R, McNeil M, Challinor S et al. The INTERNIST-1/Quick Medical Reference project: Status Report. Western Journal of Medicine, 1986, 145:816-822.

[8] ASTM. Standard Specification for Defining and Sharing Modular Health Knowledge Bases (Arden Syntax for Medical Logic Modules), ASTM, 1992.

[9] Johansson BG, Wigertz OB. An object oriented approach to interpret medical knowledge based on the Arden Syntax. Proc 16th SCAMC, McGraw-Hill, 1992; 52-56.

[10] Gao X, Shahsavar N, Arkad K, Ahlfeldt H, Hripcsak G and Wigertz O. Design and functions of Medical Knowledge editors for the Arden Syntax. In Proceedings for MEDINFO 92, 1992.

[11] Gao X, Johansson B, Shahsavar N, Arkad K, Åhlfeldt H, Wigertz O. Pre-compiling Medical Logic Modules into C++ in Building Medical Decision Support Systems. Submitted to Computer Methods and Programs in Medicine.

[12] Hripcsak G, Clayton PD, Pryor TA, Haug P, Wigertz OB, Van der Lei J. The Arden Syntax for Medical Logic Modules. Proc 14th SCAMC, IEEE Comp Soc Press, 1990; 200-204.

[13] McDonald CJ. Action-Oriented Decisions in Ambulatory Medicine. Chicago: Year Book Medical Publishers, 1981.

[14] Daltek Laboratory Data System, version 1. Daltek AB, Borlänge, Sweden.

[15] Borland C++ Compiler, version 3.0, Borland Inc., USA.

[16] Codebase ++, version 1.05, Sequiter Software Inc., Canada.

[17] dBC III Plus library, version 1.02, Lattice, Inc.. Lombard, Illinois, USA.

[18] Microsoft Editor, Microsoft Corporation, USA.

[19] Lacher DA, Connelly DP. Rate and Delta Checks Compared for Selected Chemistry Tests. Clinical Chemistry, 1988;34; 1966-1970.

[20] Gillespie K. New Standards Promise Open Database Access. Data Based Advisor, Dec. 1992, 89-93.